

NUMERICAL HILBERT FUNCTIONS FOR MACAULAY2

ROBERT KRONE

ABSTRACT. The *NumericalHilbert* package for *Macaulay2* includes algorithms for computing local dual spaces of polynomial ideals, and related local combinatorial data about its scheme structure. These techniques are numerically stable, and can be used with floating point arithmetic over the complex numbers. They provide a viable alternative in this setting to purely symbolic methods such as standard bases. In particular, these methods can be used to compute initial ideals, local Hilbert functions and Hilbert regularity.

1. INTRODUCTION

Symbolic algorithms in algebraic geometry are well studied and can be very powerful, but for many applications they can also be ineffective. Many problems are most naturally phrased over the real or complex numbers, and the running times of symbolic algorithms may become impractical. There has been a recent push to develop numerical or symbol-numeric hybrid alternatives. These methods have already shown success in tackling problems in engineering and other fields [10], for example using homotopy tracking to quickly solve zero-dimensional polynomial systems [1], [5]. Numerical methods are good at computing information about varieties, and often do best when things are regular, but we would also like insight into schemes structure.

The Macaulay dual space is a tool that can be used to numerically compute local combinatorial information, such as multiplicity structure and Hilbert function, about a polynomial system at a particular point. In conjunction with other tools, this information can be used to recover scheme theoretic data about an ideal. First introduced in [6], for a thorough discussion we refer to [8]. The *NumericalHilbert* package for *Macaulay2* [3] provides methods for both computing dual spaces and extracting information from them, implementing algorithms described in [9], [2] and [4]. These methods are designed to work using floating point arithmetic over the complex numbers, and are stable to numerical error.

Related existing software includes ApaTools by Zeng [11] and a Maple package of Mantzaflaris and Mourrain [7]. Both can compute multiplicity structure of an isolated solution to a polynomial system, while our package expands this functionality by including tools for computing on positive-dimensional varieties as well.

We will assume we have black-box methods for computing numerical rank, kernel or image of a matrix. In practice this can be done with singular value decomposition. Additionally we will assume access to a method that can approximately sample points from a variety, with any desired precision.

2. DUAL SPACE

Let $R = \mathbb{C}[x_1, \dots, x_n]$ and let R_d denote the subspace of polynomials of degree at most d . The *truncated dual space* is the vector-space dual $D_0^d = (R_d)^*$, i.e. the set of functionals $R_d \rightarrow \mathbb{C}$. The *Macaulay dual space* (or *local dual space*) of R at the origin is

$$D_0 = \bigcup_{d \geq 0} D_0^d.$$

For each monomial $x^\alpha \in R$, we define a monomial functional ∂^α by

$$\partial^\alpha \left(\sum_{\beta \in \mathbb{N}^n} c_\beta x^\beta \right) = c_\alpha.$$

Note that these monomial functionals form a basis of D_0 , and so its elements can be represented as “polynomials” in $\mathbb{C}[\partial_1, \dots, \partial_n]$.

For an arbitrary point $y \in \mathbb{C}^n$ we can also define the dual space at y , denoted D_y , by translating y to the origin. The monomial functional $\partial_y^\alpha \in D_y$ evaluates to 1 on $\prod_{i=1}^n (x_i - y_i)^{\alpha_i}$ and to 0 on all other monomials in $x_1 - y_1, \dots, x_n - y_n$. We will take our point of interest to be the origin without loss of generality, since we can make it so by a change of coordinates.

The classes `PolySpace` and `DualSpace` defined in the `NAGtypes` package are used to represent finite-dimensional subspaces of R and D_y respectively. `PolySpace` stores a basis of the subspace, and `DualSpace` stores a basis and the base point y . For convenience we express dual space elements as polynomials in R by identifying ∂^α with x^α although this is an abuse of notation.

For an ideal $I \subset R$ there is a subspace of dual functionals $D_0[I] \subset D_0$ which is the orthogonal to I .

$$D_0[I] = \{p \in D_0 \mid p(f) = 0 \text{ for all } f \in I\}.$$

We refer to $D_0[I]$ as the dual space of I , while the truncated dual space of I is $D_0^d[I] = D_0[I] \cap D_0^d$.

The dual space of I exactly characterizes the local properties of I at the origin. To make this precise, let R_0 denote the localization of R at the maximal ideal $\langle x_1, \dots, x_n \rangle$. Expressing elements of R_0 as power series, we can still evaluate them with functionals in D_0 and so define $D_0[IR_0]$.

Proposition 1. *$D_0[I]$ satisfies the following*

- $D_0[I] = D_0[IR_0]$.
- *The orthogonal to $D_0[I]$ in R_0 is exactly IR_0 .*

Computing a basis for the dual space of I immediately provides useful combinatorial data about the local behavior of I at the origin. A consequence of the above proposition is that

$$\dim_{\mathbb{C}} D_0[I] = \dim_{\mathbb{C}} (R_0/IR_0).$$

Therefore if the origin is an isolated solution to the system described by I then its multiplicity is equal to $\dim_{\mathbb{C}} D_0[I]$.

There is a simple algorithm to compute the dual space of I truncated to degree d from a generating set F . A functional $p \in D_0^d$ is in $D_0^d[I]$ if and only if $p(x^\alpha f) = 0$ for all $f \in F$ and $|\alpha| \leq d$. Form a matrix with rows the coefficient vectors of each $x^\alpha f$, including only the terms of degree $\leq d$. The kernel of this matrix consists of

the coefficient vectors of $D_0^d[I]$. For a full discussion of this algorithm we refer to [2].

The method `truncatedDual` computes a basis of the truncated dual of an ideal I at a point y . If y is known to be an isolated solution then `zeroDimensionalDual` can be used to compute a basis for the full dual space of I . The optional argument `Strategy` allows the user to specify which algorithm to use. Using `Strategy => DZ` will use the simple algorithm described above. However the default `Strategy => BM` is an implementation of an algorithm developed by Mourrain in [9], which generally has better asymptotic running time.

Example 2. Let $I = \langle x_1^2 + x_2^2 - 4, (x_1 - 1)^2 \rangle \subset \mathbb{C}[x_1, x_2]$. An approximate solution to the system is $y = (1, 1.7320508)$.

```
i1 : needsPackage "NumericalHilbert";
i2 : R = CC[x_1,x_2];
i3 : I = ideal{x_1^2 + x_2^2 - 4, (x_1 - 1)^2}
o3 = ideal (x_1^2 + x_2^2 - 4, x_1^2 - 2x_1 + 1)
o3 : Ideal of R
i4 : y = point matrix{{1,1.7320508}};
i5 : zeroDimensionalDual(y, I)
o5 = | -1 .866025x_1-.5x_2 |
o5 : DualSpace
```

Here $D_y[I] = \text{span}\{-1, 0.866025\partial_1 - 0.5\partial_2\}$ has dimension 2, the multiplicity of the solution approximated by y .

3. COMPUTING HILBERT FUNCTIONS

If I is positive-dimensional at the point of interest, the dual space of I has infinite dimension and computing a basis is not possible. However truncated duals can always be computed, the dimensions of which correspond to values of the Hilbert function. We define the *local Hilbert function* of I at 0 to be

$$H_{IR_0}(d) := \dim_{\mathbb{C}} R_0/(IR_0 + \mathfrak{m}^{d+1}) - \dim_{\mathbb{C}} R_0/(IR_0 + \mathfrak{m}^d)$$

where $\mathfrak{m} = \langle x_1, \dots, x_n \rangle$, the maximal ideal at 0.

Proposition 3. $H_{IR_0}(d) = \dim_{\mathbb{C}} D_0^d[I] - \dim_{\mathbb{C}} D_0^{d-1}[I]$.

The method `hilbertFunction` can be applied to a `DualSpace` to find its dimension in a particular degree or list of degrees.

Example 4. Consider the cyclic 4-root system $I = \langle x_1 + x_2 + x_3 + x_4, x_1x_2 + x_2x_3 + x_3x_4 + x_4x_1, x_1x_2x_3 + x_2x_3x_4 + x_3x_4x_1 + x_4x_1x_2, x_1x_2x_3x_4 - 1 \rangle$. It cuts out a curve in \mathbb{C}^4 with several singular points including $(-1, 1, 1, -1)$. We obtain an approximation y of this point with our numerical solver.

```

i6 : R = CC[x_1..x_4];

i7 : I = ideal {x_1 + x_2 + x_3 + x_4,
               x_1*x_2 + x_2*x_3 + x_3*x_4 + x_4*x_1,
               x_2*x_3*x_4 + x_1*x_3*x_4 + x_1*x_2*x_4 + x_1*x_2*x_3,
               x_1*x_2*x_3*x_4 - 1};

i8 : y = point matrix{{-1.000000000000002-3.43663806114685e-15*ii,
                       .9999999999999981+1.03073641806016e-14*ii,
                       1.000000000000002-1.08861573140903e-14*ii,
                       -.9999999999999984+3.31351409927520e-15*ii}};

i9 : D = truncatedDual(y, I, 6);

i10 : hilbertFunction({0,1,2,3,4,5,6}, D)

o10 = {1, 2, 1, 1, 1, 1, 1}

o10 : List

```

We see that values of $H_{IR_y}(d)$ for d from 0 to 6 are 1, 2, 1, 1, 1, 1, 1. We might guess from this that the Hilbert function remains constant for all $d \geq 2$ but this can't be verified from the information here so far.

For $d \gg 0$ the Hilbert function $H_{IR_0}(d)$ is polynomial in d and this is called the *Hilbert polynomial*, $HP_{IR_0}(d)$. If the Hilbert polynomial has degree r , then IR_0 has dimension $r + 1$. The multiplicity of I at the origin is $cr!$ where c is the lead coefficient of HP_{IR_0} .

To compute the Hilbert polynomial of positive-dimensional IR_0 we use the dual space to compute the generators of the initial ideal of IR_0 using the algorithm described in [4]. Let \succeq be a graded monomial order on the monomial functional of D_0 , and let \geq be the reverse order but on R_0 , i.e. a graded local order on the monomials. The initial ideal of IR_0 , denoted $\text{in}_{\succeq} IR_0$, is the monomial ideal generated by the initial terms of elements of IR_0 . Similarly the initial dual space is $\text{in}_{\succeq} D_0[I]$ spanned by the initial terms of $D_0[I]$. A monomial x^α is in $\text{in}_{\succeq} IR_0$ if and only if ∂^α is not in $\text{in}_{\succeq} D_0[I]$, or equivalently

$$\text{in}_{\succeq} D_0[I] = D_0[\text{in}_{\succeq} IR_0].$$

By observing the monomials missing from $\text{in}_{\succeq} D_0[I]$ we search for generators of $\text{in}_{\succeq} D_0[I]$. The stopping criteria described in [4] makes this search exhaustive.

The method **gCorners** produces the generators of the initial ideal $\text{in}_{\succeq} IR_0$. Similarly **sCorners** produces a list of the maximal monomials not in $\text{in}_{\succeq} IR_0$. To compute the Hilbert polynomial of IR_0 we can use the existing method **hilbertPolynomial** on the monomial ideal generated by the output of **gCorners**. The method **localHilbertRegularity** computes the degree d at which the Hilbert function and Hilbert polynomial agree.

Example 5. We continue with the cyclic 4-root system from Example 4. We omit intermediate output for readability.

```

i11 : G = gCorners(y, I)

o11 = | x_4 x_3 x_2^2 x_1x_2 |

          1      4
o11 : Matrix R  <--- R

```

```

i12 : hilbertPolynomial(monomialIdeal G, Projective=>false)
o12 = 1
o12 : QQ[i]

```

The generators of the initial ideal $\text{in}_{\geq} IR_0$ are $\{x_4, x_3, x_2^2, x_1x_2\}$. The Hilbert function of an ideal and its initial ideal agree. The Hilbert polynomial of a monomial ideal such as $\text{in}_{\geq} IR_0$ is simple to compute and here we see it is $HP_{IR_0}(d) = 1$. Since the Hilbert polynomial has degree 0, the local dimension of I at y is 1. The multiplicity is also 1.

```

i13 : localHilbertRegularity(y, I)
o13 = 2

```

The regularity of the Hilbert function is $d = 2$. This confirms that $H_{IR_y}(d) = 1$ for all $d \geq 2$. We have now completely described the local Hilbert function of I at y .

REFERENCES

- [1] Daniel J Bates, Jonathan D Hauenstein, Andrew J Sommese, and Charles W Wampler. Bertini: Software for numerical algebraic geometry. Available at www.nd.edu/~sommese/bertini, 2006.
- [2] Barry H Dayton and Zhonggang Zeng. Computing the multiplicity structure in solving polynomial systems. In *International Symposium on Symbolic and Algebraic Computation*, pages 116–123. ACM, 2005.
- [3] Daniel R. Grayson and Michael E. Stillman. Macaulay2, a software system for research in algebraic geometry. Available at www.math.uiuc.edu/Macaulay2/.
- [4] Robert Krone. Numerical algorithms for dual bases of positive-dimensional ideals. *Journal of Algebra and Its Applications*, 12(06):1350018, 2013.
- [5] Anton Leykin. Numerical algebraic geometry. *Journal of Software for Algebra and Geometry*, 3:5–10, 2011.
- [6] Francis Sowerby Macaulay. *The algebraic theory of modular systems*. Cambridge University Press, 1916.
- [7] Angelos Mantzaflaris and Bernard Mourrain. Deflation and certified isolation of singular zeros of polynomial systems. In *Proceedings of the 36th international symposium on Symbolic and algebraic computation*, pages 249–256. ACM, 2011.
- [8] M Marinari, H Möller, and T Mora. On multiplicities in polynomial system solving. *Transactions of the American Mathematical Society*, 348(8):3283–3321, 1996.
- [9] B. Mourrain. Isolated points, duality and residues. *J. Pure Appl. Algebra*, 117/118:469–493, 1997. Algorithms for algebra (Eindhoven, 1996).
- [10] Andrew J. Sommese and Charles W. Wampler, II. *The numerical solution of systems of polynomials*. World Scientific Publishing Co. Pte. Ltd., Hackensack, NJ, 2005.
- [11] Zhonggang Zeng. Apatools: a software toolbox for approximate polynomial algebra. In *Software for Algebraic Geometry*, pages 149–167. Springer, 2008.